Preprocessor Statements

Overview of Preprocessor Statements

Statements to Include Copy Code Generated from Predict file objects

The preprocessor statements COPY, FORMAT-BUFFER and GENERATE in COBOL, PL/I and Assembler programs instruct the preprocessor to include data definitions or a format buffer.

COPY
Instructs the preprocessor to insert copy code which has previously been generated by Predict. XRef data is written for the file and each field in the file.

FORMAT-BUFFER
Instructs the preprocessor to generate an Adabas format buffer and insert it at the position of the statement. XRef data is written for the file and each field in the file.

GENERATE
Instructs the preprocessor to generate a record buffer and optionally a format buffer and insert it at the position of the statement. XRef data is written for the file and each field in the file.

Statements to Write XRef Data for 3GL Copy/Include Code or Function Calls

The preprocessor statements ENTRY and CALL - if included in Assembler programs - instruct the preprocessor to write XRef data for entry points or the call of external programs. Information to be written to XRef data is specified in parameters of the statement.

CALL
Specifies the name of a called external program or function that is to be stored in the active reference records. This command can only be used for Assembler programs.

ENTRY
Specifies the name of a program entry that is to be stored in the active reference records. This command can only be used for Assembler programs.

PROGRAM
Member ID used in XRef data. This statement is interpreted only if no member name is specified when the Preprocessor is called.

Format of Preprocessor Statements

The following rules apply to preprocessor statements:

- Preprocessor statements start with EXEC ADABAS (there may be any number of blanks between the two words, but both must be coded on the same line).
- Preprocessor statements can be terminated in any of the following ways:
  - with the statement END-EXEC
  - in COBOL by a period (.)
  - in PL/I by a semicolon (;)
  - In BAL, the preprocessor terminates processing at the end of the current card unless a continuation character is punched in column 72.
- The preprocessor assumes standard statement layout in BAL, for example the Assembler statement ICTL is not valid.
- Preprocessor control statements are left in the source program as comments.

Using Keyword and/or Positional Parameters

- Parameters of preprocessor statements can be specified in positional or keyword form. Both forms are described below.
- Keyword and positional parameters can be mixed. This allows the use of a keyword parameter as a starting point for subsequent positional parameters, as shown in the following example:
  For Assembler, the INIT parameter is the eighth parameter of the EXEC Adabas GENERATE statement. VALIDATION is the next parameter in the list and can therefore be specified as a positional parameter directly behind the INIT parameter.

```
EXEC ADABAS GENERATE <file-name>,INIT=<init>,<validation>
```

CALL

The name of a called external program or function can be specified for Assembler programs. The name is stored in the active reference records.

Syntax with Positional Parameters

```
EXEC ADABAS CALL <function-name>
                        END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS CALL FUNCTION=<function-name>
                        END-EXEC
```

Parameters	
function-name	The name of a called external program or function that should be stored in the active reference records. Maximum length: 8 characters.

COPY

Instructs the preprocessor to include copy code previously generated by Predict. If more than one copy code member has been generated for the appropriate file and language, the copy code name must be specified.
XRef data is written for the file and each field in the file.

Copying a Member from a File that has been Modified after Generation

This statement can also be used if the file has been modified after the corresponding member was generated. The system behavior depends on the Predict version with which the member was generated and on parameter Ignore changes. See table below.

Syntax

with Positional Parameters

```
EXEC ADABAS COPY <file-name><copycode-name> <ignore-changes>
                     END-EXEC
```


with Keyword Parameters

```
EXEC ADABAS COPY FILE=<file-name>
                    MEM=<copycode-name>
                    IGNORE-CHANGES=<ignore-changes>
                    END-EXEC
```

Note:
All parameters except <file-name> are optional.

Parameters	
file name	The ID of the Predict file object from which the copy code was generated. See GENERATE for a list of file types that can be used for generating copy code.
member name	The member name which was supplied during copy code generation. Maximum length: 8 characters.
ignore changes	This parameter determines whether a member connected to a file that has been modified after generation is copied. **N** Default. A member connected to a file that has been modified after generation is not copied. The Preprocessor issues an error message and terminates with condition code 107. **Y** A member connected to a modified file is copied - irrespective of whether the member contains usage information. With this option, the preprocessor writes the XRef data on the basis of the current documentation of the file in Predict. A warning is given if the file object has been modified after generation, since using this option may result in inconsistent XRef data. See Additional Usage Information for Fields in a File. **X** A member of a modified file is only copied if it was generated with Predict version 3.3 or above. With this version, additional information on the usage of fields in the file is available. This information is used by the Preprocessor when writing XRef data. See Additional Usage Information for Fields in a File. If the file has been modified since generation and the usage information is not available because the member was generated with an earlier version of Predict, the Preprocessor issues an error message and terminates with condition code 107.

ENTRY

Declares the name of an entry point in Assembler programs. The name is stored in the active reference records.

Syntax with Positional Parameters

```
EXEC ADABAS ENTRY <entry-name>
                    END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS ENTRY FUNCTION=<entry-name>
                    END-EXEC
```

Parameters
entry-name The name of an entry point to be stored in the XRef data. Maximum length: 8 characters.

FORMAT-BUFFER

The FORMAT-BUFFER statement instructs the preprocessor to generate an Adabas format buffer and insert it into the 3GL member at the position of the statement. XRef data is written for the file and each field in the file.

Syntax with Positional Parameters

```
EXEC ADABAS FORMAT-BUFFER
                    <file-name><format-buffer-name>
                    <sync><offset><dcl>
                    <adabas-version><buffer-format>
                    <literal-delimiter>
                    END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS FORMAT-BUFFER
                    FILE=<file-name>
                    FORMAT-BUFFER-NAME=<format-buffer-name>
                    SYNC=<sync>
                    OFFSET=<offset>
                    DCL=<dcl>
                    ADA-VER=<adabas-version>
                    FORMAT=<buffer-format>
                    DELIMITER=<literal-delimiter>
                    END-EXEC
```

Note:
All parameters except <file-name> are optional.

Parameters	
file-name	Specifies the ID of the file object in Predict. This parameter is mandatory. The file can be one of the following types: • Adabas file (type A), where parameter Adabas SQL usage may not be set to Y. • Adabas userview (type U)
format-buffer-name	The name to be given to the format buffer in the target program. In COBOL, this may be up to 30 characters long; in PL/I, up to 31; and in BAL, up to 8 characters. The default value is the same as in the respective generation function.
sync	**Y** All appropriate fields will be aligned. **N** No fields will be aligned. **S** Fields will be aligned only if the corresponding Predict object has the 3GL specification Synchronized=S.
offset	**L,Y,P** The total length of the code will be included in the format buffer. **V** A constant will be generated for the format buffer length.
dcl	Only applies to PL/I include code. **Y** The generated code will be preceded by a declare statement DCL and will end with a semicolon instead of a comma.
adabas-version	The version of Adabas for which the copy code of the Adabas files and userviews is to be generated. See list of possible values in the section Adabas Version.
buffer-format	**Y** Normal format for format buffer. Adabas groups, standard formats and lengths are used whenever possible. The resulting format buffers are then as short as possible. **F** Full format for format buffer. The format buffer will include field length and format.
literal-delimiter	Only valid for COBOL copy code. **S** single quotes **D** double quotes

GENERATE

The statement GENERATE instructs the preprocessor to generate copy code from Predict file objects and insert it into the 3GL member at the position of the statement. XRef data is written for the file and each field in the file.

Generation can be performed for Assembler, COBOL and PL/I. See table of valid file types and languages below.

The generated copy code will be written to the source area and then copied into the source program.

Code	File Type	Assembler	COBOL	PL/I
A	Adabas file	Y	Y	Y
B	Adabas SQL view	Y	Y	Y
BT, BV	Adabas D table/view		Y	
D, E	DB2 table/view	Y	Y	Y
F	rdb file	Y	Y	Y
J	IMS segment layout	Y	Y	Y
JT, JV	INGRES table/view		Y	Y
L	logical VSAM file	Y	Y	Y
M	ISAM file	Y	Y	Y
O	other file	Y	Y	Y
OT, OV	ORACLE table/view		Y	Y
S	sequential file	Y	Y	Y
T	rms file	Y	Y	Y
U	Adabas user view	Y	Y	Y
V	VSAM file	Y	Y	Y
X	General SQL file	Y	Y	Y
XT, XV	INFORMIX table/view		Y	
YT, YV	SYBASE table/view		Y	
1	LEASY	Y	Y	Y
2	ISAM BS2000	Y	Y	Y

Generating Assembler Copy Code

Syntax with Positional Parameters

```
EXEC ADABAS GENERATE <filename><prefix><suffix><dsect>
                     <dc-ds><nr-comments><offset><init>
                     <validation><truncation><dsect-name><align>
                     <equ><adabas-version><generate-format-buffer>
                     <format-buffer-name><counter-length>
                     END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS GENERATE FILE=<file-name>
                     PREFIX=<prefix>
                     SUFFIX=<suffix>
                     DSECT=<dsect>
                     DC-DS=<dc-ds>
                     NR-COMMENTS=<nr-comments>
                     OFFSET=<offset>
```



```
INIT=<init>
VALIDATION=<validation>
TRUNCATION=<truncation>
RECORD-BUFFER-NAME=<dsect-name>
SYNC=<align>
EQU=<equ>
ADA-VER=<adabas-version>
FORMAT-BUFFER=<generate-format-buffer>
FORMAT-BUFFER-NAME=<format-buffer-name>
COUNTER-LENGTH=<counter-length>
END-EXEC
```

Note:
All parameters except <file-name> are optional.

Details of the parameters are given in the following table and also in the section Generation in this documentation. If a parameter is omitted, the default value defined in Predict by the DDA will be used. See the section Defaults in the **Predict Administration documentation**.

If the statement is entered using positional parameters and no prefix or no suffix is required, an asterisk (*) should be substituted for the parameter <prefix> or <suffix>.

Parameters	
file-name	ID of the Predict file object from which the Assembler copy code is to be generated. This parameter is mandatory. See table of valid file types in the description of Parameter Generate.
prefix	A prefix for the field names. Maximum length: 8 characters.
suffix	A suffix for the field names. Maximum length: 8 characters.
dsect	**Y** The copy code will be generated as an ASSEMBLER DSECT (dummy section). The DSECT will have the name specified by the <dsect-name> parameter, or the file ID if no <dsect-name> parameter is supplied.
dc-ds	**DC** Assembler DC (define constant) instructions will be generated. **DS** Assembler DS (define storage) instructions will be generated.
nr-comments	Specifies the number of abstract lines per field (0-16) which will be included in the generated code.

offset	**Y** Include the offset of each item in the record buffer structure (relative to the beginning of the structure) in decimal and hexadecimal formats as a comment. The total length of each buffer is also included. **P** Include the absolute position (offset+1) as a comment. **L** Include the total lengths of the record buffer and the format buffer as a comment. **V** Only allowed if parameters As DSECT=N and With DC or DS=DC. The file number and the calculated lengths of the record buffer and the format buffer are to be generated as constants in the copy code. The name of the file number constant is the record buffer name with N as prefix. The name of each length constant is the appropriate buffer name with L as prefix. Each name is prefixed, suffixed, validated and truncated in the same way as any other field name. **N** No offset.
init	This option takes effect only when With DC or DS=DC and As DSECT is set to N. **N** No initialization. **Y** Statements are generated to initialize the structure with the value specified for Init value in the corresponding field object in Predict. Fields with no value for Init value are initialized with zeros or blanks. Aligned 8-digit fields with format B or I are not initialized. In a PE group with the 3GL specification Gr.structur set to blank, only the first occurrence of each field is initialized.
validation	Determines how invalid characters in a field name are handled. **blank** Invalid characters will result in an error message but will not be deleted. **rep.char.** Invalid characters will be replaced by this character. Valid values: letters A-Z, digits 0-9, $, § or #. ***** Invalid characters will be deleted.
truncation	Specifies which characters are deleted if a generated field name is longer than 8 characters: **L** truncate from the left **R** truncate from the right **M** truncate from the middle.

dsect-name	Specifies the name of the record buffer in the generated structure. The effect of this parameter depends on parameter dsect.
align	**Y** All appropriate fields will be aligned. **N** No fields will be aligned. **S** Fields will be aligned only if the corresponding Predict object has the 3GL specification Synchronized= S.
equ	**Y** EQU statements are to be generated for fields of length 1 whose format is not P and comment lines are to be generated for other fields, using any condition names defined as attributes of the Predict field objects. These names are prefixed, suffixed, validated and truncated in the same way as field names. For fields of format L where no condition name was specified, a condition name is generated by concatenating the field name "Example" to the prefix N. In this case the following statement is generated: NEXAMPLE EQU X’00’.
adabas-version	The version of Adabas for which the copy code of the Adabas files and userviews is to be generated. See list of possible values in the section Adabas Version.
generate-format-buffer	Format buffer generation for Assembler copy code is only allowed if parameters As DSECT=N and With DC or DS=DC. The contents of the format buffer will correspond exactly to the contents of the record buffer. Only valid for files of type A (with parameter Adabas SQL usage set to N) or for files of type U. Valid values: **Y** Adabas format buffer is to be generated. Adabas groups, standard formats and lengths are used whenever possible. The resulting format buffers are then as short as possible. **F** Full format buffer is to be generated. Length and format of Adabas fields are included. **N** No format buffer is to be generated. **Note:** If you are generating for a WANG environment, set this parameter to F or N.
format-buffer-name	Specifies the label (name) of the format buffer in the generated structure. If omitted, the file ID prefixed by F is used.
counter-length	Length of additional counter fields: Valid values: 1, 2.

Generating COBOL Copy Code

Syntax with Positional Parameters

```
EXEC ADABAS GENERATE <file-name><prefix><suffix> <start-level>
                     <level-increment><shift-number><nr-comments>
                     <offset><init><validation><truncation>
                     <record-buffer-name><cond-name><sync>
                     <indexed><depending>
                     <adabas-version><generate-format-buffer>
                     <format-buffer-name>
                     <check-name>
                     <literal-delimiter>
                     <decimal-char>
                     <redefine-name>
                     END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS GENERATE FILE=<file-name>
                     PREFIX=<prefix>
                     SUFFIX=<suffix>
                     START-LEVEL=<start-level>
                     LEVEL-INCREMENT=<level-increment>
                     SHIFT-NUMBER=<shift-number>
                     NR-COMMENTS=<nr-comments>
                     OFFSET=<offset>
                     INIT=<init>
                     VALIDATION=<validation>
                     TRUNCATION=<truncation>
                     RECORD-BUFFER-NAME=<record-buffer-name>
                     COND-NAME=<cond-name>
                     SYNC=<sync>
                     INDEXED=<indexed>
                     DEPENDING=<depending>
                     ADA-VER=<adabas-version>
                     FORMAT-BUFFER=<generate-format-buffer>
                     FORMAT-BUFFER-NAME=<format-buffer-name>
                     CHECK-NAME=<check-name>
                     DELIMITER=<literal-delimiter>
                     DEC-CHAR=<decimal-char>
                     REDEFINE-NAME=<redefine-name>
                     END-EXEC
```

Note:
All parameters except <file-name> are optional.

Details of the parameters are given in the following table and also in the section Generation in this documentation. If a parameter is omitted, the default value defined in Predict by the DDA will be used. See the section Defaults in the **Predict Administration documentation**.

If the statement is entered using positional parameters and no prefix or no suffix is required, an asterisk (*) should be substituted for the parameter <prefix> or <suffix>.

Parameters	
file-name	ID of the Predict file object from which the COBOL copy code is to be generated. This parameter is mandatory. See table of valid file types in the description of Parameter Generate.

prefix	A prefix for the field names. Maximum length: 16 characters.
suffix	A suffix for the field names. Maximum length: 16 characters.
start-level	Specifies the starting level of the generated record buffer. Valid values are in the range 1 - 40.
level-increment	Specifies the level-increment. Valid values are in the range 1 - 40.
shift-number	The number of positions to be shifted right when a level number which is higher than the current level number is encountered. Valid values are in the range 0 - 9.
nr-comments	Specifies the number of abstract lines per field that will be included in the generated code. Valid values are in the range 0 - 16.
offset	**Y** The offset of each item in the record buffer structure (relative to the beginning of the structure) in decimal and hexadecimal formats is to be included as a comment. The total length of each buffer is also included. **P** As above, but the absolute position (offset+1) is included as a comment. **L** The total lengths of the record buffer and the format buffer are to be included as a comment. **V** The file number and the calculated lengths of the record buffer and the format buffer are to be generated as constants in the copy code. The name of the file number constant is the record buffer name prefixed by N-. The name of each length constant is the appropriate buffer name prefixed by L-. Each name is prefixed, suffixed, validated and truncated in the same way as any other field name. **N** No offset.
init	**Y** The fields will be initialized wherever possible using a COBOL VALUE clause. Any fields with INIT VALUEs in their Predict objects will be initialized with those values; other fields will be initialized with low values (zeros or spaces). **S** Only fields with INIT VALUEs in the corresponding Predict object will be initialized. **N** No initialization.

validation	Determines how invalid characters in a field name are handled. **blank** Invalid characters will result in an error message but will not be deleted. **rep.char.** Invalid characters will be replaced by this character. Valid values: letters A-Z, digits 0-9 or hyphen. ***** Invalid characters will be deleted.
truncation	Specifies which characters are deleted if a generated field name is longer than 30 characters: **L** truncate from the left **R** truncate from the right **M** truncate from the middle.
record-buffer-name	Specifies the name of the record buffer in the generated structure. If omitted, the file ID is used.
cond-name	**Y** Any condition names defined in the Predict field objects are to be generated on level 88, provided that the respective field objects have one of the following formats: • **A** All lengths • **N** or **P** Less than 19 digits • **I** or **B** 2, 4 or 8 digits • **L** A FALSE-condition will always be generated. The Condition name is then generated by concatenating the field name to the prefix N- (if not specified explicitly). These names are prefixed, suffixed, validated and truncated in the same way as field names.
sync	**Y** All appropriate fields will be aligned. **N** No fields will be aligned. **S** Fields will be aligned only if the corresponding Predict object has the 3GL specification Synchronized = S.

indexed	**Y** The COBOL clause INDEXED BY will be generated for all repetitive fields (MU/MC and PE/PC). **S** This clause will be generated only for repetitive fields which have INDEXED BY NAMEs in their Predict objects. **N** This clause will not be generated for any field.
depending	**Y** COBOL attribute OCCURS DEPENDING ON is generated for a field or field group in a file if it has type PE or MU and DEPENDING ON NAME is specified for this field. These names are prefixed, suffixed, validated and truncated in the same way as field names. **Note:** This option is not allowed for files of type A or U. This option is ignored when using a WANG COBOL compiler.
adabas-version	The version of Adabas for which the copy code of the Adabas files and userviews is to be generated. See table of valid values in the section Adabas Version.
generate-format-buffer	The contents of the format buffer will correspond exactly to the contents of the record buffer. Only valid for files of type A (with parameter Adabas SQL usage set to N) or for files of type U. Valid values: **Y** Adabas format buffer is to be generated. Adabas groups, standard formats and lengths are used whenever possible. The resulting format buffers are then as short as possible. **F** Full format buffer is to be generated. Length and format of Adabas fields are included. **N** No format buffer is to be generated. **Note:** If you are generating for a WANG environment, set this parameter to F or N.
format-buffer-name	Specifies the name of the format buffer in the generated structure. If omitted, the file ID prefixed by FORMAT-BUFFER- is used.
check-name	**A** COBOL field names are checked for uniqueness throughout the whole structure. **Y** Structure levels are included in the validation check of the field names: if two fields have the same name, they must be separated by at least one field with a different name and a lower-level number. **N** No check for duplicate field names is performed

literal-delimiter	**S** single quotes, **D** double quotes
decimal-character	**P** point, **C** comma
redefine-name	Determines how COBOL field names for Predict fields of type RE are generated: **F** The string FILLER is used as redefinition name. **S** The suffix REGR is added to the Predict field ID. If a field is redefined more than once, the suffix will have the form REGRn, where n is an integer incremented by 1 for each field of type RE.

Generating PL/I Include Code

Syntax with Positional Parameters

```
EXEC ADABAS GENERATE <file-name><prefix><suffix>
                     <start-level><level-increment><shift-number>
                     <nr-comments><offset><init><struct-as-char>
                     <static><validation><truncation>
                     <record-buffer-name><align><dcl>
                     <adabas-version><generate-format-buffer>
                     <format-buffer-name><check-name>
                     <numeric sign><position of sign>
                     END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS GENERATE FILE=<file-name>
                     PREFIX=<prefix>
                     SUFFIX=<suffix>
                     START-LEVEL=<start-level>
                     LEVEL-INCREMENT=<level-increment>
                     SHIFT-NUMBER=<shift-number>
                     NR-COMMENTS=<nr-comments>
                     OFFSET=<offset>
                     INIT=<init>
                     STRUCTURE=<struct-as-char>
                     STATIC=<static>
                     VALIDATION=<validation>
                     TRUNCATION=<truncation>
                     RECORD-BUFFER-NAME=<record-buffer-name>
                     SYNC=<align>
                     DCL=<dcl>
                     ADA-VER=<adabas-version>
                     FORMAT-BUFFER=<generate-format-buffer>
                     FORMAT-BUFFER-NAME=<format-buffer-name>
```



```
CHECK-NAME=<check-name>
NUM-SIGN=<numeric sign>
POS-SIGN=<position of sign>
END-EXEC
```

Note:
All parameters except <file-name> are optional.

Details of the parameters are given in the following table and also in the section Generation in this documentation. If a parameter is omitted, the default value defined in Predict by the DDA will be used. See the section Defaults in the **Predict Administration documentation**.

If the statement is entered using positional parameters and no prefix or no suffix is required, an asterisk (*) should be substituted for the parameter <prefix> or <suffix>.

Parameters	
file-name	ID of the Predict file object from which the PL/I include code is to be generated. This parameter is mandatory. See table of valid file types in the description of Parameter Generate.
prefix	Specifies a prefix for the field names. Maximum length: 16 characters.
suffix	Specifies a suffix for the field names. Maximum length: 16 characters.
start-level	Specifies the starting level of the generated record buffer. Valid values are in the range 1 - 40.
level-increment	Specifies the level-increment. Valid values are in the range 1 - 40.
shift-number	The number of positions to be shifted right when a level number which is higher than the current level number is encountered. Valid values are in the range 0 - 9.
nr-comments	Specifies the number of abstract lines per field that will be included in the generated code. Valid values are in the range 0 - 16.

offset	**Y** The offset of each item in the record buffer structure (relative to the beginning of the structure) in decimal and hexadecimal formats is to be included as a comment. The total length of each buffer is also included. **P** As above, but the absolute position (offset+1) is included as a comment. **L** The total lengths of the record buffer and the format buffer are to be included as a comment. **V** The file number and the calculated lengths of the record buffer and the format buffer are to be generated as constants in the include code. The name of the file number constant will be the record buffer name prefixed by N_. The name of each length constant will be the appropriate buffer name prefixed by L_. Each name is prefixed, suffixed, validated and truncated in the same way as any other field name. **N** No offset.
init	**Y** The fields will be initialized wherever possible. Any fields with INIT VALUEs in their Predict objects will be initialized with those values; other fields will be initialized with low values (zeros or spaces). **S** Only fields with INIT VALUEs in the corresponding Predict object will be initialized. **N** No initialization.
struct-as-char	**Y** The entire generated structure will be declared at the end of the record buffer as a single character-string.
static	**Y** The structure will be declared with the attribute STATIC.
validation	Determines how invalid characters in a field name are handled. **blank** Invalid characters will result in an error message but will not be deleted. **rep.char.** Invalid characters will be replaced by this character. Valid values: letters A-Z, digits 0-9, $, §, # or _ (underscore). ***** Invalid characters will be deleted.

truncation	Specifies which characters are deleted if a generated field name is longer than 31 characters: **L** truncate from the left **R** truncate from the right **M** truncate from the middle.
record-buffer-name	Specifies the name of the record buffer in the generated structure. If omitted, the file ID is used.
align	**Y** All appropriate fields will be aligned. **N** No fields will be aligned. **S** Fields will be aligned only if the corresponding Predict object has the 3GL specification Synchronized= S. **Note:** This parameter only takes effect with fields that have the PL/I attribute FIXED BIN or FLOAT DEC.
dcl	**Y** The generated code will be preceded by a declare statement DCL and will end with a semicolon instead of a comma.
adabas-version	The version of Adabas for which the include code of the Adabas files and userviews is to be generated. See table of valid values in the section Adabas Version.
generate-format-buffer	The contents of the format buffer will correspond exactly to the contents of the record buffer. Only valid for files of type A (with parameter Adabas SQL usage set to N) or for files of type U. Valid values: **Y** Adabas format buffer is to be generated. Adabas groups, standard formats and lengths are used whenever possible. The resulting format buffers are then as short as possible. **F** Full format buffer is to be generated. Length and format of Adabas fields are included. **N** No format buffer is to be generated. **Note:** If you are generating for a WANG environment, set this parameter to F or N.
format-buffer-name	Specifies the name of the format buffer in the generated structure. If omitted, the file ID prefixed by FORBUF_ is used.

check-name	**A** Field names are checked for uniqueness throughout the whole structure. **Y** Structure levels are included in the validation check of the field names: if two fields have the same name, they must be separated by at least one field with a different name and a lower level number. **N** No check for duplicate field names is performed
numeric sign	Specifies which of the PL/I picture characters T, I or R is to be used for the representation of numeric values of format packed with sign or unpacked with sign.
position of sign	Defines the position of the sign in a numeric field: **L** left **R** right.

PROGRAM

The member name can be passed to the preprocessor with the PROGRAM statement:

Syntax with Positional Parameters

```
EXEC ADABAS PROGRAM <member-name><library>
                    END-EXEC
```

Syntax with Keyword Parameters

```
EXEC ADABAS PROGRAM
                    PROGRAM-ID=<member-name>
                    LIBRARY-ID=<library>
                    END-EXEC
```

Note:
If member and library are specified when the Preprocessor is called, these values are taken. The statement EXEC ADABAS PROGRAM is then not necessary.

Parameters	
member-name	The name used to identify the XRef data. Maximum length: 8 characters.
library	If the parameter library is specified, a system of type G (3GL application) that contains this library name in its implementation pointer must have been defined before. If no library is specified, the *SYSCOB*, *SYSBAL* or *SYSPLI* libraries are used.

